PHG4TrackFastSim
Fast Tracking using GenFit and PHG4Hit

Jin Huang(BNL),

Haiwang Yu (NMSU)

Sep 06, 2016

Introduction

Motivation: Quickly produce fsPHENIX tracking performance with Geant simulation + Kalman Filter.

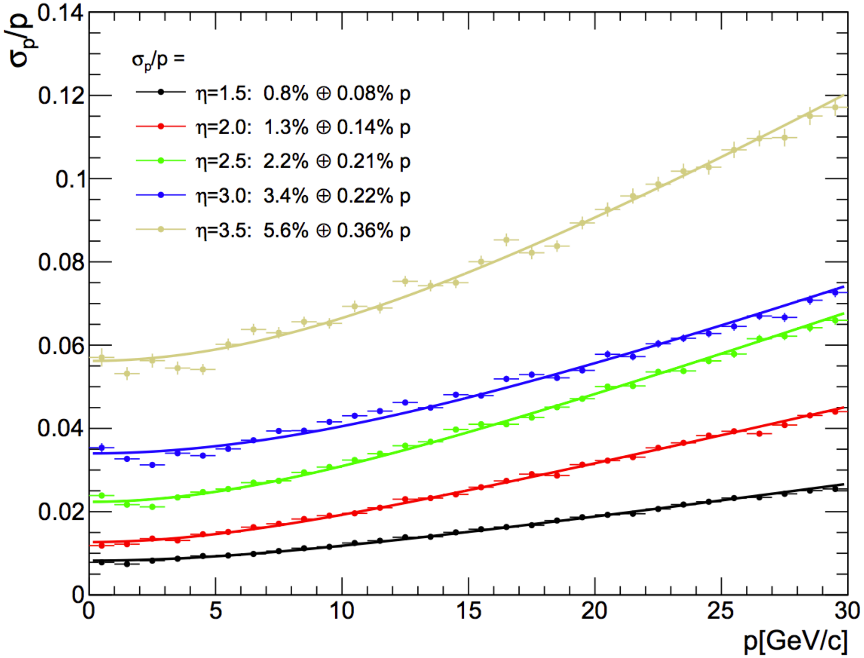
- More realistic than Sagitta calculation.
- Good estimation before the detector design finalized.
- Serves as prototype for forward sPHENIX tracking software
  - with future pattern recognition component.

Procedure:

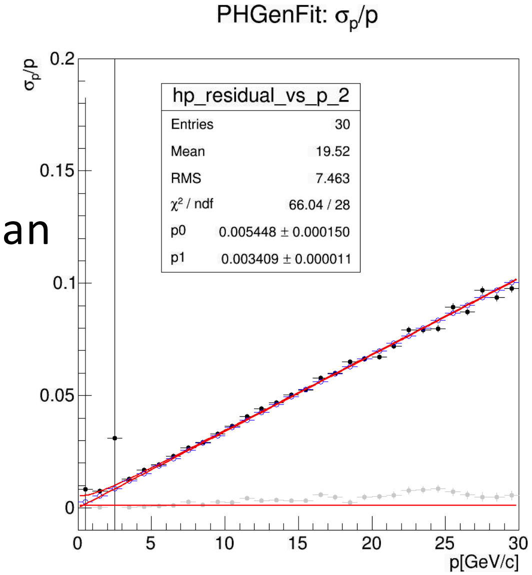
- Simulation
  - ⟹ PHG4TruthInfoContainer.
  - ⟹ PHG4Hit.
  - ⟹ TGeo detector geometry in DST run node (PHGeometry by Jin)
- Feed the Kalman Filter:
  - PHG4Hit ⟹ Measurements: Smear PHG4Hit according to given detector resolution.
  - Measurements grouping: use MC truth information, "pseudo pattern recognition"
  - Seed: Smeared MC truth information
- Output:
  - SvtxTrackMap ⟹ SvtxTrack_FastSim, contains truth track index for simplified truth track matching.
- SpinFest2016 Slides:
  - https://indico2.riken.jp/indico/getFile.py/access?contribId=19&resId=0&materialId=slides&confId=2284

In previous talks:

Resolution vs. pT for different eta

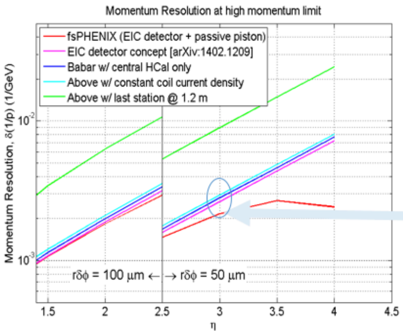


Sagitta vs. Full Kalman



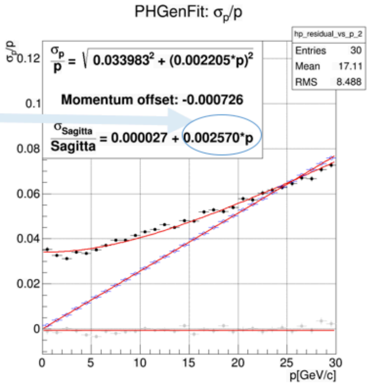
Compare to previous Sagitta Calculation

Jin's calculation based on vertex + optimum Sagitta plane + 300cm last station.



GenFit fitting for η = 3.0, corresponding to magenta curve in left plot.

- The linear term, p1, from the σ_S/S **is consistent with left plot,** both are ~0.25%.
- The p1 term from full GenFit Kalman is better than σ_S/S . That could be caused by that we have more stations in full Kalman.

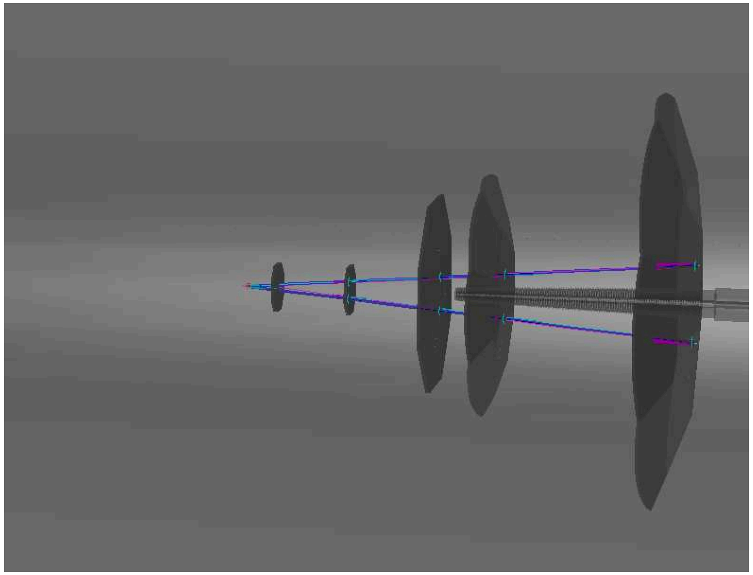


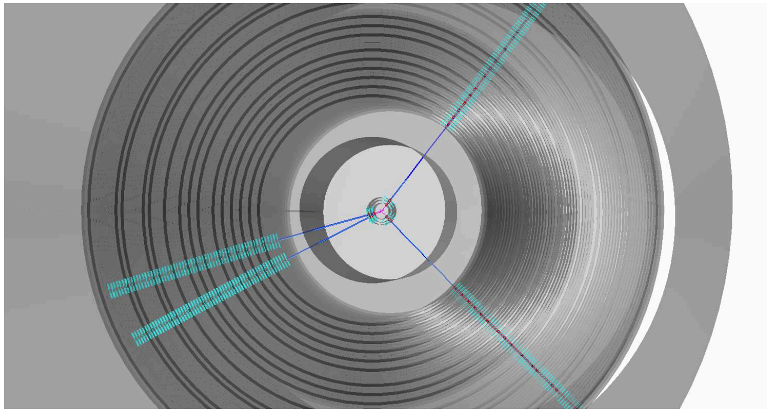
Structure

- coresoftware/g4hough/PHG4TrackFastSim:
  - Merged Pull Request #192
  - Tracking module
  - Input: PHG4TruthInfoContainer and PHG4HitsContainer
  - Output: SvtxTrackMap with SvtxTrack_FastSim filled in.

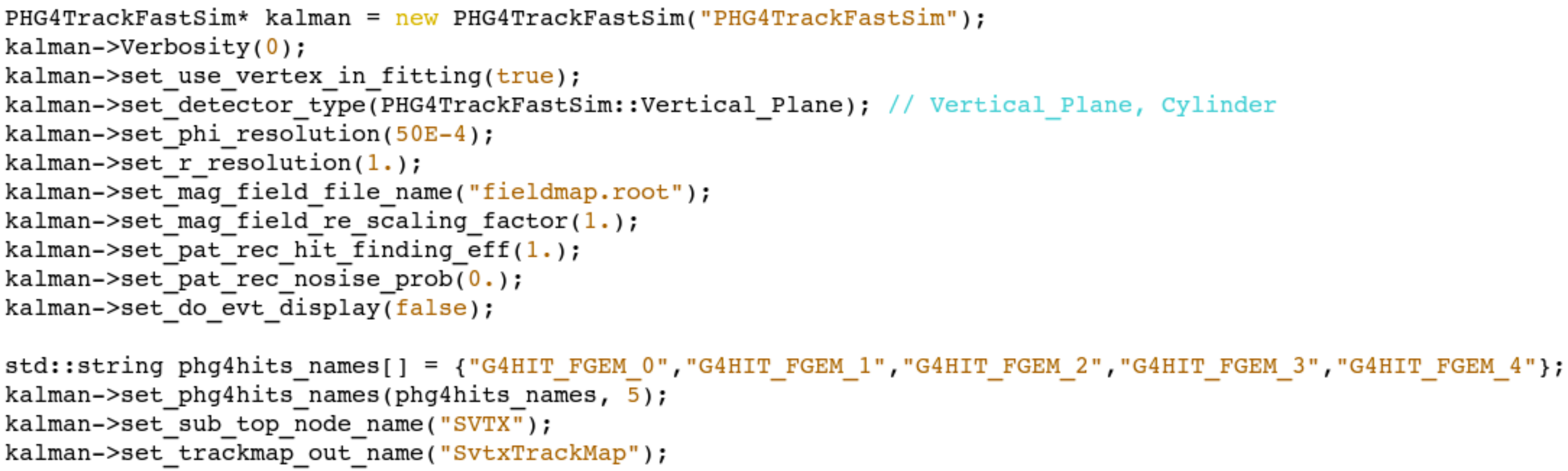
- analysis/Tracking/FastTrackingEval:
  - Fill eval NTuples and Histos

- example macros:
  - analysis/Tracking/FastTrackingEval/macros





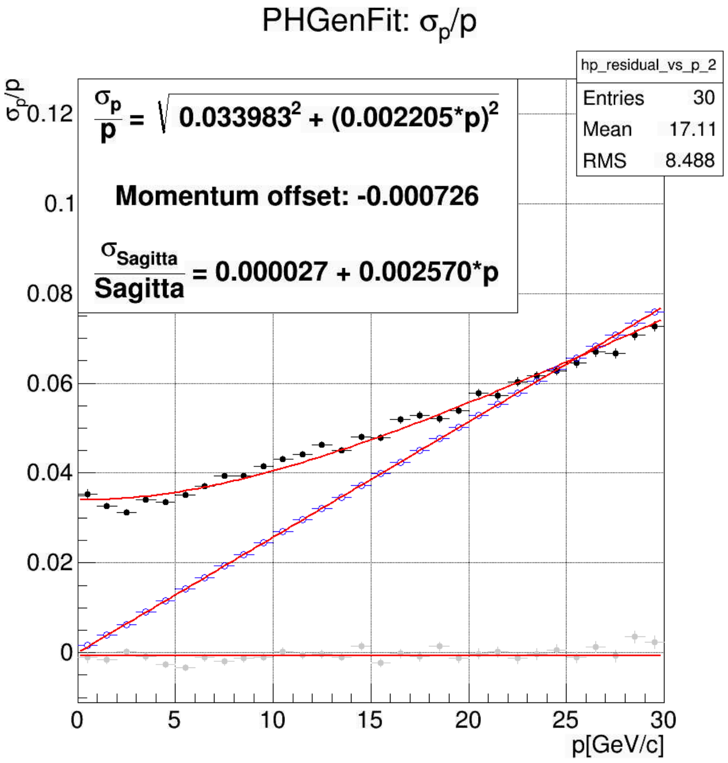
Switches



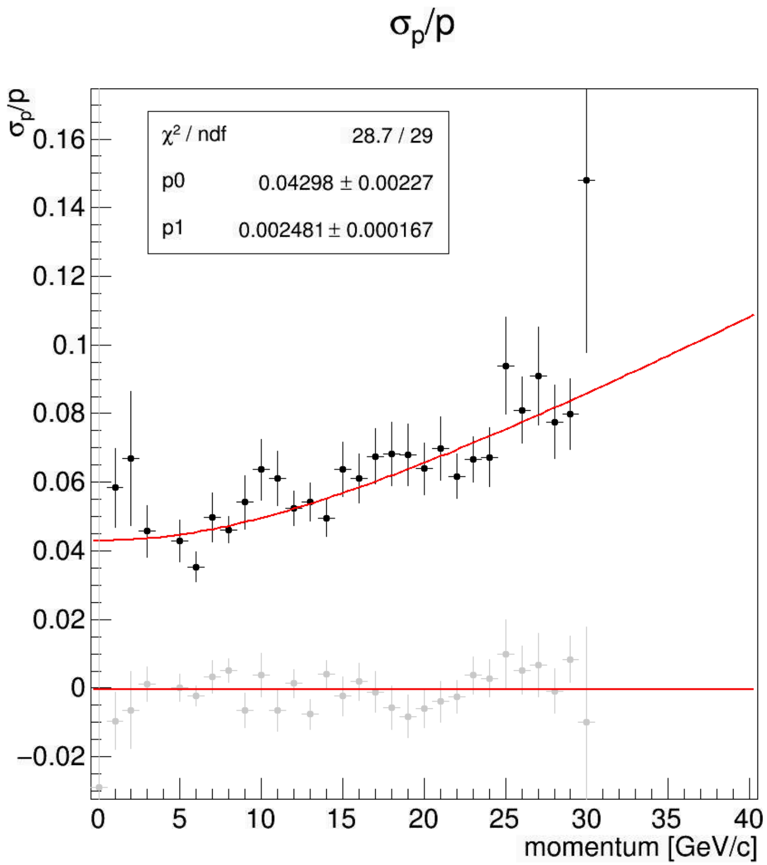
```
PHG4TrackFastSim* kalman = new PHG4TrackFastSim("PHG4TrackFastSim");
kalman->Verbosity(0);
kalman->set_use_vertex_in_fitting(true);
kalman->set_detector_type(PHG4TrackFastSim::Vertical_Plane); // Vertical_Plane, Cylinder
kalman->set_phi_resolution(50E-4);
kalman->set_r_resolution(1.);
kalman->set_mag_field_file_name("fieldmap.root");
kalman->set_mag_field_re_scaling_factor(1.);
kalman->set_pat_rec_hit_finding_eff(1.);
kalman->set_pat_rec_nosise_prob(0.);
kalman->set_do_evt_display(false);

std::string phg4hits_names[] = {"G4HIT_FGEM_0","G4HIT_FGEM_1","G4HIT_FGEM_2","G4HIT_FGEM_3","G4HIT_FGEM_4"};
kalman->set_phg4hits_names(phg4hits_names, 5);
kalman->set_sub_top_node_name("SVTX");
kalman->set_trackmap_out_name("SvtxTrackMap");
```

Comparing with previous standalone code

Standalone program



This module



To-dos

More tests with this module.

Investigate some generic pattern recognition options.

- e.g. from OLYMPUS Experiment